GRANULARITY METRICS FOR IT SERVICES

Completed Research Paper

Bernd Heinrich
University of Regensburg
Universitätsstraße 31,
93053 Regensburg, Germany
bernd.heinrich@wiwi.uni-regensburg.de

Steffen Zimmermann
University of Innsbruck
Universitätsstraße 15,
6020 Innsbruck, Austria
steffen.zimmermann@uibk.ac.at

Abstract

Many articles discuss the design of fine- or coarse-grained IT services without defining what granularity exactly means. However, this is important to know since the granularity influence e.g. the reusability or the composition effort of services and thus the development, maintenance and composition costs of a service realization. Therefore, we present different metrics to evaluate and compare the granularity of various service realizations of a process where each metric represents a different perspective on service granularity. In addition, we discuss the advantages and disadvantages of these metrics and argue that a reasonable measurement of service granularity can usually be obtained by the combination of the metrics. Furthermore, we illustrate that the proposed metrics may constitute a first step towards the evaluation of the total costs of various service realizations. The application of the metrics as well as the discussion of the practical benefits is illustrated by a real world case.

Keywords: IT Service, Service Granularity, Granularity Metrics, Service-oriented Architecture, Design Science

Introduction

In the context of Service-oriented Architectures (SoA), which are currently establishing as leading IT architectural paradigm in theory and practice, the design of IT services[1] is a key issue to reach the *goal*[2] of efficiently supporting business processes. Thereby, many scholars and practitioners emphasize the *question* how to adequately choose the service granularity (cf. Erl 2004; Krafzig et al. 2005) and recognize this as the crucial question in designing IT services. This is due to the fact that service granularity influences important aspects like the reusability of IT services or the performance of the execution of a service composition (e.g. Alahmari et al. 2010; Haesen et al. 2008). The economic effects resulting from the choice of service granularity are discussed for instance by the following trade-off (Erl 2004): The more coarse-grained IT services are, the less they may be reused and the more they may actually impose redundant functions or multiple versions. This leads to higher implementation and maintenance costs compared to fine-grained IT services. In contrast, fine-grained IT services have a higher potential of reuse. However, if multiple fine-grained IT services are used to execute a process, their composition is getting more complex which leads to much higher composition costs compared to coarse-grained IT services. This economic trade-off between implementation and maintenance costs on the one hand and composition costs on the other hand motivates the importance of measuring service granularity.

Although many publications on SoA refer to fine- versus coarse-grained IT services, it is hardly discussed how the service granularity and thus the resulting implementation, maintenance and composition costs can be measured. Many authors define (often implicitly) service granularity as the extent of functions realized by a service (e.g. Erl 2007; Marks and Bell 2006; Papazoglou and van den Heuvel 2006; Galster and Bucherer 2008). We follow this general understanding of service granularity in this paper. Thus, we delimit other types of granularity like the so called “business value granularity” or the “data granularity” (Haesen et al. 2008). In addition, we further delimit design criteria such as cohesion or coupling (e.g. Aier 2006; Perepletchikov et al. 2007a and 2007b; Albani et al. 2008; Schelp and Winter 2008). These criteria and associated design principles like “minimizing coupling” and “maximizing cohesion” are typically discussed and applied for an adequate design of services (as well as for object-oriented modules (Chidamber and Kemerer 1994), components, etc.). However, the metrics to measure cohesion and coupling (e.g. Counsell et al. 2006; Feuerlicht 2011; Perepletchikov et al. 2010 and 2011) do not explicitly aim to measure service granularity in terms of the extent of functions that are realized by an IT service.

Against this backdrop, we present in this paper different formally defined *metrics* to measure and compare the granularity of various service realizations of a process. In addition, we analyze the advantages and disadvantages of these metrics. In this respect, the granularity metrics presented below *complement* the large number of existing procedures for the function-oriented (e.g. Aier 2006; Albani et al. 2008; Her et al. 2008; Schelp and Winter 2008) or economic-oriented (e.g. Krammer et al. 2011; Wang et al. 2005) identification and design of services (or components). This means, our metrics do not substitute these procedures but can be applied independently of the chosen decomposition procedure. Rather, the metrics allow a more precise statement what is meant by fine- or coarse-grained IT services and constitute a basis to estimate the implementation, maintenance and composition costs of different service realizations.

The paper is based on the Design Science Research Paradigm. Thereby, we particularly base our work on the seven guidelines for conducting Design Science Research by Hevner et al. (2004). After the brief discussion of the general relevance of the problem within this introduction (“problem identification and motivation”), we review existing literature that deals with the measurement of service granularity and identify a research gap (“define the objectives for a solution”). Then we develop iteratively four artifacts in the sense of granularity metrics (“design and development”) and analyze their corresponding advantages and disadvantages. Previously, we introduce the so called function and service graph to be able to define those metrics technically. The practical applicability (“demonstration”) of the artifacts as well as its practical

---

[1] An IT service is referred to as a software artifact that comprises an IT realization of functions and an interface with particular characteristics (cf. Buhl et al. 2008; Krafzig et al. 2005; Papazoglou et al. 2008; Papazoglou 2003). Our metrics are technology neutral and may not only be used to measure the granularity of web services, for instance.

[2] According to the Goal-Question-Metric approach of Basili et al. (1994) for developing a meaningful metrics program top-down, we indicate the three components of this approach by italic words in the introduction.

utility ("evaluation") is extensively illustrated by a real world case of a large German bank. Finally, the last section summarizes the results regarding the guidelines of Hevner et al. (2004) and suggests fields for further research ("communication").

Literature review

To provide an overview on the extent to which service granularity is treated in the literature and to create a basis to develop our metrics, we conducted an extensive review of the literature which is briefly described in this section following the guidance of Webster and Watson (2002).

To *identify the relevant literature*, we started with a scan of the journals within the AIS Senior Scholars' Basket (cf. http://home.aisnet.org/displaycommon.cfm?an=1&subarticlenbr=346) and complemented this with first and foremost design oriented journals (e.g. Decision Support Systems, Business & Information Systems Engineering, and several ACM journals) and conference proceedings (e.g. International Conference on Information Systems, European Conference on Information Systems, and several IEEE conferences) of the Information Systems and Computer Science disciplines. Furthermore, we conducted a keyword search using the Google Scholar service (http:\\scholar.google.com). There, we searched for all possible combinations of the terms service, web service, and component in combination with the term granularity (and their corresponding plural forms). Afterwards, we went backwards by reviewing the citations for the thitherto identified articles to determine prior considerable articles. Finally, we went forward by using again the Google Scholar service to identify articles citing the previously found articles. The main contributions on service granularity which we could identify in this way we will discuss below.

To *structure the results of our review*, we distinguish the identified articles in two sets: (1) A large set of articles which use the term service granularity without defining a formal metric for measuring granularity. (2) A small set of articles which define formal metrics to measure the service granularity. In the following, we first discuss the literature of set (1) and afterwards the literature of set (2).

Set (1) consists primarily of contributions to design or identify services based on functional criteria (Aier 2006; Albani et al. 2008; Erl 2007; Fiege and Stelzer 2007; Her et al. 2008; Winkler 2007; Winter 2003; Schelp and Winter 2008). These approaches usually include an implicit definition of the term service granularity. For example, Aier (2006) interprets service granularity as the number of functions realized by a service. Fiege and Stelzer (2007) have a similar interpretation but further consider the abstraction level of a service. For this purpose, they decompose the considered domain top down. In other words, they define different decomposition layers for the functions. Thus, the abstraction level is determined based on the decomposition layer where the realized function is located. Winkler (2007) also performs a decomposition of functions and implicitly measures the service granularity in the same way. Winter (2003) and Schelp and Winter (2008) interpret service granularity as the size of the realized functions. However, in all these papers no formal metric to measure the service granularity is explicated.

In contrast, the articles of set (2) comprise formal metrics to determine service granularity (cf. Haesen et al. 2008; Khoshkbarforoushha et al. 2010; Ma et al. 2009; Sims 2005; Wang et al. 2006; Zhang and Li 2009): The simplest metrics are described by Sims (2005) and Haesen et al. (2008) who both count the number of functions realized by one service (by these metrics the interpretation of Aier (2006) and partially of Fiege and Stelzer (2007) is formalized). Haesen et al. (2008) further distinguish between the so called *Default Functionality Granularity* (number of functions provided as default by a service) and the *Parameterized Functionality Granularity* (number of functions which are optionally provided by a service and which can be parameterized). Both granularities are defined as absolute values from the perspective of the service user. This interpretation of service granularity is extended by the approach of Wang et al. (2006). Based on the fact that functions (so called "features") can be composed and decomposed, they define a formal tree structure to represent decomposition layers of functions. In a first step, Wang et al. (2006) define "a feature *f*'s granularity *G(f)* as the sum of all its child features granularities" and explicate this in a technically defined metric. The granularity metric of Wang et al. (2006) is defined as $G(f) = \begin{cases} \sum_{f_i \in child(f)} G(f_i), & if\ child(f) \neq \Phi \\ 1 & if\ child(f) = \Phi \end{cases}$ where Φ is a function which cannot be further decomposed. By this metric, the granularity is also measured as an absolute value which represents the number of functions implemented by a service. In a second step, Wang et al. (2006) consider so called "granularity layers" which refers to the decomposition layers of the formal tree structure. Depending on which decomposition layer a realized function is located, the more fine- or coarse-grained the associated software artifact (e.g. component or service) is assessed. However, they do not integrate the decomposition layers in the previously defined formal metric. Furthermore, a comparison between granularities of different trees is limited because of the missing normalization of the metric values. Ma et al. (2009) also decompose processes, activities and functions. However, they determine the granularity based on the set of all implemented services (so called "service portfolio"). Accordingly, the granularity of a single service is calculated as the average number of realized activities or functions over all services of the service portfolio. However, if the number of realized activities or functions differs strongly among the single services, the measurement is imprecise. Zhang and Li (2009) and Khoshkbarforoushha et al. (2010) measure the granularity of a *composed service*. According to the publications of Haesen et al. (2008) and Sims (2005), they simply calculate this granularity as the number of services which are directly and indirectly composed by a service. The different implicit and explicit definitions of service granularity which are used in the discussed articles of the two sets of literature are summarized in Table 1.

Table 1. Selected approaches on defining service granularity

Set (1): …not explicating a formal metric		*Set (2): …explicating a formal metric*	
Articles	*Granularity definition*	*Articles*	*Granularity definition*
Aier (2006)	Number of functions realized by a service	Haesen et al. (2008) Sims (2005); Wang et al. (2006)	Number of functions realized by a service
Winkler (2007)	Decomposition layer of a service	Ma et al. (2009)	Average number of realized functions over all services
Fiege and Stelzer (2007)	The number of realized functions and the decomposition layer of a service	Zhang and Li (2009); Khoshkbarforoushha et al. (2010)	Number of services which are directly and indirectly composed by a service
Winter (2003); Schelp and Winter (2008)	Size of functions realized by a service		

To sum up, in literature some formal metrics are already proposed. These metrics mostly consider the *number of realized functions*. Furthermore, it is occasionally mentioned that the *decomposition layer* (i.e. where a realized function is located within a formal function tree; e.g. Wang et al. 2006) and the *size of the realized functions* (e.g. Schelp and Winter 2008) also affect the service granularity. However, a formal metric that takes both or at least one of the latter criteria explicitly into account is missing so far in literature. In addition, the service granularity is measured as an absolute value by the existing metrics. *Normalized metric values* would enhance the interpretation and comparison of the metric values of different services. To close this research gap, we aim to answer the following research question.

Research Question: How may the service granularity be measured taking into account the previously discussed criteria *number of realized functions, decomposition layer, size of the realized functions,* and *normalized metric values*?

To answer this research question we iteratively design four granularity metrics. Furthermore, we give an interpretation of the values of the metrics and analyze the advantages and disadvantages of these metrics.

Measuring service granularity

Some of the above mentioned function-oriented approaches define rules to decompose functions, identify and assign services to the decomposed functions and represent the results of both tasks in a graph. We use such an existing graph definition - which is largely inspired by the well-known approach of Lee et al. (2001) - in order to complement these function-oriented approaches and to have an adequate basis to develop the granularity metrics (cf. Krammer et al. 2011). This means, we refer to both existing rules to decompose functions and an existing graph definition. This minimizes not only the subjectivity regarding the creation of a particular graph resulting from the modelers' preferences but also enhances the significance of the service granularities measured based on such a graph. In the following, we call this graph a function and service graph (FSG). The FSG represents the decomposition relations among the functions as well as the assignment of services to the directly realized functions. A formal definition of the FSG is explicated in the following subsection.

Function and Service Graph

The FSG is a directed, acyclic graph $G=(N, E)$ with a set of nodes N and a set of directed edges E. The set of functions M are represented by nodes ($M \subseteq N$) and a disaggregation relation $(m_i, m_j) \in E$ between the functions $m_i \in M$ and $m_j \in M$ is represented by a directed edge. The decomposition of a function m_i into the functions $m_j, \ldots, m_{j+n}$ is disjoint and complete. A directed edge (m_i, m_j) means "function m_j is part of function m_i". Every source node of the FSG, which means, a node without any incoming edge, is called a *process* $p \in P$ (with $P \subseteq N$ as the set of all processes). Every sink of the FSG, which means, a node without any outgoing edge, is called a *basic function* $m_b \in B$ (with $B \subseteq M$ as the set of all basic functions). The inner nodes of the graph that are neither processes nor basic functions, are called *preceding functions* $m_v \in V$ (with $V \subseteq M$ as the set of all preceding functions). A sequence of nodes and edges $p, (p, m_1), \ldots, (m_{b-1}, m_b)$, m_b is called a *path* with the beginning node p and the end node m_b. If a path begins with a process (i.e. $p \in P$) and ends with a basic function (i.e. $m_b \in B$), it is called a *complete path*. The *length* $d(p, m_b)$ of a path $w(p, m_b)$ is defined as the number of functions $m_i \in M$ of the path w. The shortest (longest) path from a node p to a node m_b is the path where the length $d(p,m_b)$ is minimal (maximal). Additionally, every service $s_i \in S$ is also are represented by a node (with $S \subseteq N$ as the set of all services) and is assigned to exactly one function $m_j \in M$ by the directed edge $(s_i,m_j) \in E$. A service realizes the function m_j completely, which means, all functions m_k, where a path $w(m_j, m_k)$ exist, are included. A function m_j where $(s_i, m_j) \in E$ holds, is called a *realized function*.

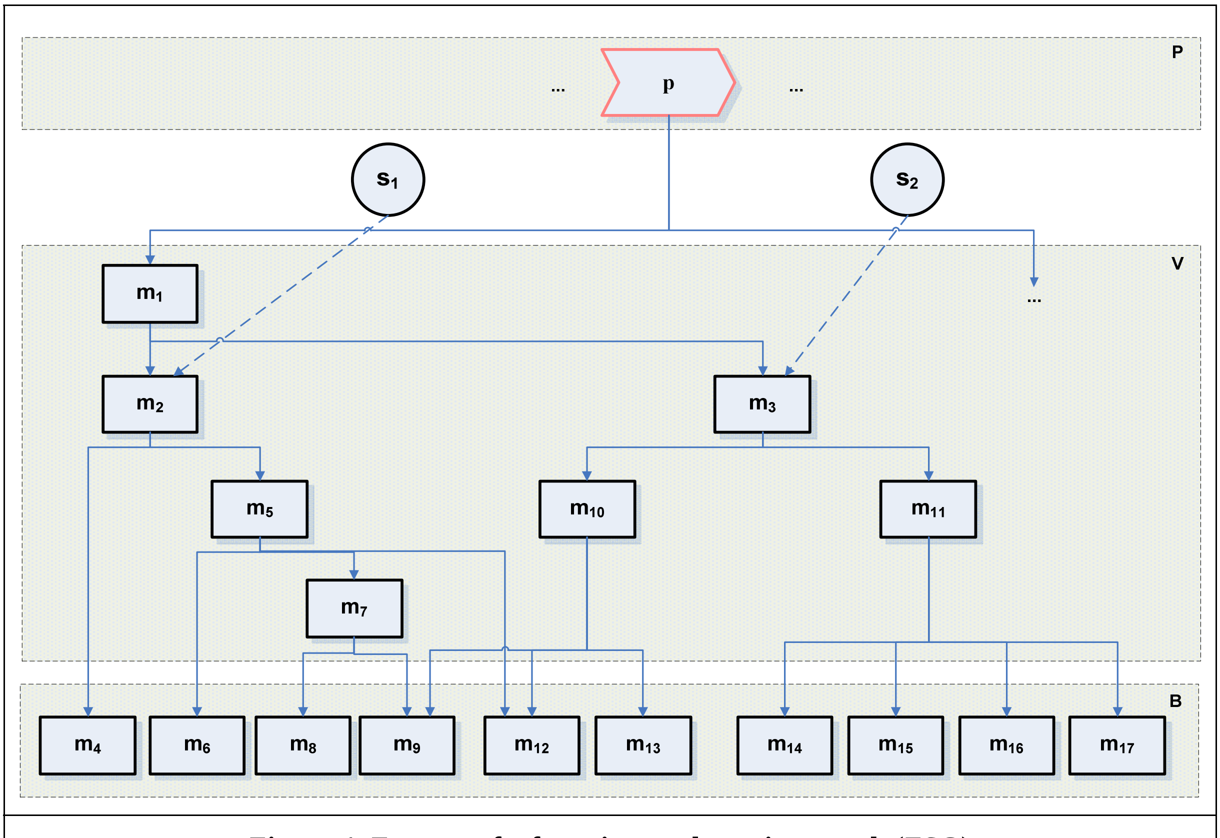

Figure 1. Extract of a function and service graph (FSG)

An extract of a FSG is exemplified in Figure 1 which we will use to illustrate the granularity metrics in the following section: As result of a functional analysis this FSG represents the decomposition of a process p. The function m_1, which is required to compose p, is decomposed into the functions m_2 and m_3 which are further decomposed on their part. Additionally, it becomes clear that the basic function m_9 is part of two different functions m_7 and m_{10} and that m_{12} is part of m_5 and m_{10}. Furthermore, a service realization is illustrated by the services s_1 and s_2. For example, the service s_1 realizes the function m_2 directly and thus also the preceding functions m_5 and m_7, and the basic functions m_4, m_6, m_8, m_9, and m_{12}. By this service realization the basic functions m_9 and m_{12} are realized redundantly by the services s_1 and s_2.

Granularity Metrics

In the following, we develop - based on the criteria defined in the previous section - four granularity metrics that realize different perspectives on granularity. These metrics help us to specify statements like "functions are realized by fine- or coarse-grained services". To exemplify the metrics, we calculate the value of each metric according to the FSG presented in Figure 1. In addition, the advantages and disadvantages of the four different metrics are discussed.

Width Metric

First, we present a width metric that is based on the number of *basic* functions that are directly and indirectly realized by a service. Thus, this metric is based on the definitions of service granularity by Haesen et al. (2008), Sims (2005), and Wang et al. (2006). Let b_j be the number of all basic functions realized directly and indirectly by a service s_i. Thereupon, we define the width metric as follows:

(1) $$z_w = \frac{1}{b_j} \quad \text{with } b_j > 0; \exists (s_i, m_j) \in E\ .$$

The domain of the metric is normalized to the interval [0;1]. A metric value of *one* represents the realization of a basic function (maximal fine-grained service), a value near *zero* denotes a coarse-grained realization, and a value of *zero* denotes that a service s_i do not refer to any function m_j (i.e. $\nexists (s_i,m_j) \in E$). Thus, the width metric contributes to the existing metrics in literature (cf. Haesen et al. 2008; Sims 2005; Wang et al. 2006) by *normalizing the metric values*.

Exemplifying the width metric, the granularities of the services s_1 and s_2 (see Figure 1) can be calculated as follows: According to service s_1 and s_2 the number of directly or indirectly realized basic functions is $b_j = 5$ (m_4, m_6, m_8, m_9, m_{12}) and $b_j = 7$ (m_9, m_{12}, m_{13}, m_{14}, m_{15}, m_{16}, m_{17}), respectively. This leads to a granularity of $z_w = 0.2$ for service s_1 and $z_w = 0.14$ for service s_2. Hence, service s_1 is more fine-grained than service s_2 in terms of the width metric.[3]

To measure the overall granularity of a FSG in terms of the width metric, the ratio of the total number of realized functions to the total number of basic functions in the FSG can be used (this is similar to the granularity defined by Ma et al. 2009). This overall granularity can easily be interpreted since a value of the metric greater than one means that at least one basic function is realized redundantly. In addition, this overall granularity is easy to calculate in practice, since the metric does not need the granularities of

---

[3] An alternative width metric could be defined as the number of the directly following functions of the realized function m_j in relation to the maximal number of the following functions over all complete paths. This ratio has to be subtracted from 1 to ensure that a value of the metric of 1 represents again the realization of a maximal fine-grained service and a value of 0 denotes a maximal coarse-grained realization. In the exemplified FSG, the number of the directly following functions of the realized function m_2 for service s_1 equals 2 (m_4 and m_5). The maximal number of the following functions over all complete paths equals 3 (m_6, m_7 and m_{12}) for function m_5. Therefore, a granularity of 0.33 results from subtracting the ratio of 2/3 from 1. Analogously, a granularity of 0.5 results for service s_2. As this alternative metric takes only directly following functions into account, service s_2 is more fine-grained than service s_1 (although service s_2 realizes more basic functions than service s_1). As this result of the alternative width metric is elusive, we propose the width metric explicated in equation (1).

all single services. However, the domain of the metric is not normalized to the interval [0;1].[4]

Besides the easy interpretability the width metric has the characteristic to be independent of the number of the decomposition layers of the FSG. This aspect allows a comparison of different functional-oriented approaches on service identification and their respective FSG. For this reason, Wang et al. (2006) compared different service realizations by means of a similar metric. However, this characteristic comes along with the disadvantage that the width metric does not consider the lengths of the paths and the decomposition layers in the FSG.

Depth Metric

By the depth metric we address this disadvantage of the width metric and consider the decomposition layers of the FSG. Hence, this metric indicates the “position” of a realized function with respect to the complete path. Specifically, the depth metric is defined by the length of the path from the process to the realized function in relation to the length of the complete path. Let t_j be the number of functions $m_i \in M$ of the path $w(p, m_j)$ with p as process and m_j as realized function. Let further be t_b the number of functions $m_i \in M$ of the complete path $w(p, m_b)$ with p as process and m_b as basic function. Thereby, the path $w(p, m_j)$ is part of the complete path $w(p, m_b)$. Based on these notations, we define the depth metric as follows:

(2) $$z_d = \frac{t_j - 1}{\max[1, t_b - 1]} \text{ with } \exists (s_i, m_j) \in E$$

Using this metric, we achieve the desired results: The realization of a basic function (that does not follow directly after a process) results in a value of 1 (maximal fine-grained service), while the value of 0 (maximal coarse-grained service) results from a realized function that follows directly after a process p. To get the desired metric values, we subtract 1 from the variables t_j and t_b. In addition, the maximum out of 1 and t_b-1 in the denominator is necessary, since otherwise the metric is not defined for a realized basic function that follows directly after a process. In so doing, the domain of z_d is normalized to the interval [0;1]. In addition, if a service s_i do not refer to any function m_j (i.e. $\nexists (s_i, m_j) \in E$), the value of the metric z_d is defined as 0. If the realized function m_j is part of several complete paths $w(p, m_b)$, the metric has to be applied to the longest complete path. Thus, the depth metric is the first granularity metric that takes the *decomposition layer* where a realized function is located into account.

By means of this depth metric we can calculate the granularity of the service s_1 (see Figure 1) that realizes function m_2: The longest complete paths, in which m_2 is included, are the paths $w(p, m_8)$ and $w(p, m_9)$ both with a length of $d(p, m_n) = 5$. For both longest complete paths $t_j = 2$ (i.e. m_1 and m_2) and $t_b = 5$ (i.e. m_1, m_2, m_5, m_7 *and* m_8 *or* m_9) holds. Hence, we get a granularity for service s_1 of $z_d = \frac{2-1}{5-1} = 0.25$. Likewise, for service s_2 that realizes function m_3 results a granularity value of $z_w = 0.33$.[5]

The higher value of service s_2 compared to service s_1 is comprehensible, as the longest complete path including m_3 ($t_b = 4$) is shorter than the longest complete path including m_2 ($t_b = 5$). Thus, according to the

---

[4] To get a normalized value of the metric the arithmetic mean of the granularities z_w of all single services can be calculated. However, the resulting granularity value can hardly be interpreted and is not that easy to calculate. Thus we propose the calculation of the overall granularity like illustrated above.

[5] As an alternative to this calculation, the granularity z_d could first be calculated for each complete path where the realized function is part of. Afterwards, the arithmetic mean of all granularities z_d could be calculated. As not only the longest complete path is considered within this calculation, but all complete paths where the realized function m_j is part of, the resulting value of the metric gains information content. However, the resulting value of the metric is hardly interpretable. Using the calculation proposed above based on the longest complete path, the position of the realized function concerning its depth within the graph can be deduced directly from the value of the metric. For instance, the value of the metric of $z_d = 0.25$ indicates that the realized function is located after ¼ of the complete path. Thus, the results of the metric calculated based on the longest complete path can be compared in a cardinal way (in contrast to the arithmetic mean over all complete paths). Therefore, we propose the calculation of the depth metric based on the longest complete path.

depth metric, s_2 is more fine-grained than s_1 (in contrast to the width metric). This is reasonable, because on the one hand the function realized by service s_1 is part of a longer complete path (depth metric). On the other hand, service s_1 realizes a smaller amount of basic functions than service s_2 (width metric). Thus, the calculated granularity value of a service depends on the applied metric and thus on the perspective adopted by a user and has to be interpreted accordingly.

Besides this granularity measurement of single services, the overall granularity of all services in the FSG is also of interest. In this respect, the arithmetic mean of the granularities z_d of the single services s_i can be determined. Thus, the domain of the overall granularity is also normalized to the interval [0;1]. An overall granularity of 1 means, that only basic functions (maximal fine-grained services) are realized. In contrast, if only functions are realized that follow directly a process, the overall granularity results in 0 (maximal coarse-grained).

The depth metric is intuitively interpretable and takes – in contrast to the width metric - the decomposition layer where a service is located into account. However, it has the disadvantage that the values of the metric only depend on the length of the paths and the width of the FSG is not taken into account. This suggests that the combination of both metrics could be advantageous.

Combined Width and Depth Metric

By the combined width and depth metric, we aim to contribute to an improved measurement by integrating the advantages of the previously developed metrics. Let n_j be the number of all functions which follow directly and indirectly after the realized function m_j. Let further be n_p the sum of n_j and the number of functions which follow directly and indirectly after the process p on the path $w(p, m_j)$. Thereupon, we define the combined width and depth metric as follows:

$$(3) \qquad z_{wd} = 1 - \frac{n_j}{\max\left[1, n_p - 1\right]} \quad \text{with} \quad \exists (s_i, m_j) \in E$$

As the number of functions has to be determined starting from the process p, we subtract 1 from n_p in the denominator. In addition, the maximum out of 1 and (n_p-1) has to be calculated. To ensure that the granularity of 1 can be interpreted as maximal fine-grained, we subtract the quotient $\frac{n_j}{\max\left[1, n_p - 1\right]}$ from 1. Thus, the domain of the metric is normalized to the interval [0;1]. If a basic function is realized directly by a service, it follows $n_j = 0$ and $z_{wd} = 1$ (maximal fine grained). In case of realizing a function that follows directly after the process, it results $n_j = max[1,n_p-1]$ and therefore a granularity of $z_{wd} = 0$ (unless the realized function represents a basic function). In addition, if a service s_i do not refer to any function m_j (i.e. $\nexists (s_i,m_j) \in E$), the value of the metric z_{wd} is defined as 0. If the realized function m_j is part of several paths $w(p, m_j)$, n_p is calculated for the longest path. Thus, the combined width and depth metric is the first granularity metric that takes both, the *number of realized functions* and the *decomposition layer* where a realized function is located into account.

When exemplifying the combined width and depth metric, then for service s_1 results $n_j = 7$ (m_4 to m_9 and m_{12}), $n_p = 9$ (m_1, m_2, m_4 to m_9 and m_{12}) and thus a granularity value of $z_{wd} = 0.125$. Similarly, the granularity value for service s_2 is $z_{wd} = 0.1$. [6]

---

[6] Alternatively, n_p could be defined as the maximal number of the directly and indirectly following functions of all processes p within the FSG. By using this definition for calculating the granularities of the services s_1 and s_2, $n_p = 17$ holds for both services. Thus, we get a granularity for service s_1 of $z_{wd} = 0.56$ and for service s_2 of $z_{wd} = 0.44$. By these values of the metric, service s_1 is more fine-grained than service s_2, too. However, these values of the metric are elusive concerning the positions of the realized functions within the FSG and the number of indirectly realized basic functions. Based on the metric values one would intuitively expect that the realized functions are located in the middle of the FSG concerning width and depth. Moreover, further alternatives to calculate n_p based on different parts of the graph are conceivable. However, the greater the considered part of the graph, the higher the values of the metric for services which realize multiple basic functions and for which the realized function nearly follows the process. As high values of the metrics are an indicator for a fine-grained realization these values are hard to interpret. Thus, we propose to calculate the metric values like illustrated above.

Both granularity values are low, which means both services s_1 and s_2 can be interpreted as coarse-grained. There are two reasons for this: First, both services realize functions which are located quite at the beginning of their complete paths (depth metric). Second, both services realize functions that are followed directly and indirectly by a larger number of further functions. In terms of the combined width and depth metric, service s_1 is slightly more fine-grained than service s_2. This means in the example that the effect of the larger number of directly and indirectly realized functions by service s_2 (compared to service s_1) outweighs the effect that the function m_3 realized by service s_2 is part of a shorter complete way (compared to service s_1).

To calculate the overall granularity of the FSG the arithmetic mean of the granularities z_{wd} of each single service can be used (cf. depth metric).

The combined width and depth metric takes the path position of a realized function in the FSG (cf. depth metric) as well as the number of the functions that are directly and indirectly realized by a service (cf. width metric) into account. Thus, more information about the FSG is used by the combined width and depth metric. Therefore, the metric enables a comparison of different service realizations of the same FSG. For example, such service realizations are discussed by means of a case study of a financial services provider by Winkler (2007). Here, the presented FSG illustrates not only several decomposition relations among functions but also multiple realizations of functions. For instance, the function "determination of the root of a function" has decomposition relations to exactly three preceding functions and is further decomposed into multiple basic functions. Such decompositions may result in an asymmetric FSG (see also the real world case below), for which the application of the combined width and depth metric seems to be suitable. However, the granularity values of the combined width and depth metric cannot be interpreted as easy as the values of the width and depth metric.

All the metrics discussed so far are mainly based on the number of functions and their decomposition relations. However, if the considered functions differ strongly in their size the usage of these metrics is not sufficient. Thus, we need to design a further metric that takes the *size* of a realized function into account.

Size Metric

Regarding our size metric, we follow Winter (2003) as well as Schelp and Winter (2008) who refer to the size of functions realized by a service in the context of service granularity but do not define a formal metric. The size of a software function is often measured in lines of code (LOC) or function points. For software size measurement, different procedures exist in literature (e.g., Park 1992, Dolado 2000). However, we aim to define our size metric independently of the chosen procedure to measure the size of a software function.

Let $size_{m_j}^{agg}$ be the aggregated size of a realized function m_j with $size_{m_j}^{agg} > 0$. If m_j is a preceding function, $size_{m_j}^{agg}$ has to be calculated as the sum of the single sizes of the directly following functions m_f (equals the disjoint and complete decomposition) and the composition specification of m_j. The latter one includes first and foremost the specification of the interactions and messages between the directly following functions as well as the states and flow logic of their coordination (e.g., similar to standards like WS-BPEL; cf. Erl 2007). If m_j is a basic function, $size_{m_j}^{agg}$ equals simply the size of this single basic function. Let further be m_j part of the complete path $w(p, m_b)$ with $p, (p, m_a), \ldots, (m_{b-1}, m_b), m_b$. Based on these notations, we define the size metric as follows:

$$(4) \qquad z_s = 1 - \frac{size_{m_j}^{agg}}{size_{m_a}^{agg}} \quad \text{with } size_{m_j}^{agg}, size_{m_a}^{agg} > 0; \exists (s_i, m_j) \in E$$

If the realized function m_j follows directly a process p, m_j equals m_a and consequently the value of the metric z_s equals 0 (maximal coarse-grained). In contrast, if the realized function m_j is a basic function which do not follow directly after a process, the quotient decreases and consequently the value of the metric z_s increases. If a service s_i do not refer to any function m_j (i.e. $\nexists (s_i, m_j) \in E$), the value of the metric z_s is defined as 0. Thus, the domain of the metric is normalized to the interval [0;1[. If a realized function m_j is part of multiple paths $w(p, m_j)$, we choose the path with the maximal value for $size_{m_a}^{agg}$. Thus, the size metric is the first granularity metric that takes the *size of the realized function* into account.

To calculate the granularities for the services s_1 and s_2 of our example, we use the values for $size_{m_f}$ given by Table 2 (the values of the preceding functions comprise only the composition specification). For service s_1 we get an aggregated size of the realized function m_2 of $size_{m_2}^{agg}$ = 5,700 (sum of the sizes of the functions m_2, m_4 to m_9 and m_{12}). In addition, the aggregated size of the function which follows directly after the process m_1 results in $size_{m_1}^{agg}$ = 5,900 (sum of the sizes of the functions m_1, m_2, m_4 to m_9 and m_{12}). Consequently, we get a granularity value for service s_1 of z_s = 0.034. In doing equivalent calculations for service s_2 we get a value of the size metric of z_s = 0.025.

Table 2. Not aggregated sizes of the functions of the exemplified FSG

function	$size_{m_f}$	function	$size_{m_f}$	function	$size_{m_f}$	function	$size_{m_f}$
m_1	200	m_6	1,000	m_{11}	400	m_{16}	1,000
m_2	200	m_7	200	m_{12}	1,000	m_{17}	1,000
m_3	200	m_8	1,000	m_{13}	1,000		
m_4	1,000	m_9	300	m_{14}	1,000		
m_5	300	m_{10}	300	m_{15}	1,000		

Thus, both services s_1 and s_2 can be interpreted as coarse-grained whereupon service s_1 is more fine-grained than service s_2. The different results compared to the depth metric are due to the fact that only the size of the functions is considered and not the position of the realized function within the FSG. In our example, the size of the directly and indirectly realized functions of service s_1 is lower than of service s_2. To calculate the overall granularity of the FSG the arithmetic mean of the granularities z_s of each single service can be used like before.

The size metric may add value in addition to the previously defined metrics if the sizes (usually measured by LOC, object points or function points) of the directly and indirectly realized functions differ strongly and if these sizes are indicators for the implementation effort. However, the results of the size metric are hard to interpret. Furthermore, it does not explicitly consider the *number of realized functions* as well as the *decomposition layer* where a realized function is located.

Comparison of the Granularity Metrics

In the following, we discuss which *criteria* are considered explicitly by each granularity metric and to what extent the resulting metric values can be compared as well as interpreted (cf. Table 3).

First, the domain of each metric is normalized to the interval between 0 (maximal coarse-grained service) and 1 (maximal fine-grained service). Based on this normalization, it is possible to compare the granularity values of different services assigned to one or more FSGs. Existing metrics represent the service granularity as absolute values (see e.g. Wang et al. 2006). The advantage of a normalization becomes clear, if we assume two services realizing two functions that are located in second position after a process. Considering this position and representing service granularity as an absolute value, we would conclude the same granularity for both services. However, if these functions are part of two complete paths that differ strongly regarding their length, a normalized metric like the depth metric leads to totally different granularity values for the associated services. Hence, by normalizing the position, which means to divide the second position of both functions by the length of the corresponding complete path, relevant additional information (in this context, the length of the complete path) is taken into account. Second, a combination of the metrics is reasonable, which is easy to demonstrate by means of the following example: Assuming again two services where each is realizing five basic functions, both services have the same granularity in terms of the width metric (several works analyzed in the literature review would argue like this). However, if we further know that one service has, for example, a granularity value of 0 with respect to the depth metric (i.e. the service realizes a function that follows after the process), whereas the second service has a value of 0.8 (i.e. the service realizes almost a basic function), then the first statement has to be put into perspective. Insofar, we recommend the combination of the metrics.

Table 3. Summary of the results

	Width metric		*Depth metric*		*Combined width and depth metric*		*Size metric*	
Values of the metric	s_1	s_2	s_1	s_2	s_1	s_2	s_1	s_2
Service granularity	0.2	0.14	0.25	0.33	0.125	0.1	0.034	0.025
Overall granularity	0.2		0.29		0.11		0.03	
Considered criteria								
Normalized metric values	+		+		+		+	
Number of realized functions	+		-		+		**o**	
Decomposition layers	-		+		+		**o**	
Size of the realized function	-		-		-		+	
Interpretability and Comparability								
Interpretability	+		+		**o**		-	
Comparability	+		+		+		+	

+ considered explicitly, **o** considered implicitly, **-** not considered

In the exemplified FSG (see Figure 1), the granularities of both services s_1 and s_2 as well as the overall granularity were measured as relatively coarse grained by all metrics. This is comprehensible because both services realize multiple functions and are located - according to the complete path - in second position after the process. In addition, service s_1 is considered as more coarse-grained than service s_2 in terms of the depth metric, while the other metrics come to the opposite result. This is due to the different criteria that are taken into account by the metrics. For instance, the depth metric takes the decomposition layer where a realized function is located into account, whereas the width metric considers the number of realized functions. This issue is interesting, because in research and practice it is often stated that one service is more coarse- or fine-grained than another service without specifying the perspective on granularity, which means on which criteria or metric such a statement is based. Thus, a statement like "service s_1 is more coarse-grained than service s_2" is difficult to understand and probably insufficient without explicating the corresponding perspective.

Based on our analyses, it seems to make sense to use the combined width and depth metric as this metric takes both criteria of the depth as well as the width metric into account. In contrast, the depth and the width metric are easier to be interpreted. In this respect, we can make the following recommendations:

- If the numbers of functions realized by the considered services are nearly the same, the application of the depth metric is reasonable (as applying the width metric would result in no further information).
- If the lengths of the paths with respect to the considered services are nearly the same, the application of width metric is reasonable.
- In all other cases, the application of the combined width and depth metric is reasonable.

Furthermore, it can be valuable to combine the metrics, which is also illustrated by the real world case in the following section. The exclusive use of these three metrics is sufficient only if the sizes of the functions of the FSG do not differ strongly. Otherwise, the size metric should be used additionally. However, as the size metric cannot be interpreted easily, we suggest a combination with at least one of the three other metrics. Such a combination may be also reasonable when estimating the economic effects of the granularity choice. The implementation costs depend first and foremost on the sizes of the realized functions (cf. cost estimation models like CoCoMo (Boehm et al. 2000) which are also based on the size measured by LOC). Therefore, the calculated values of the size metric seem to be adequate to estimate the implementation costs of a service (cf. Krammer et al. 2011). In contrast, the composition costs mainly depend on the number of services which have to be composed to execute a process. Therefore, the calculated values of the width metric or the combined width and depth metric seem to be adequate to support the estimation of the composition costs.

Application of the metrics in a real world case

In this section, we present a real world case of a project at a major European bank to examine the feasibility and applicability of the granularity metrics “in depth in business” (Hevner et al. 2004, p. 86). In this context, we address the guidelines for conducting Design Science Research of problem relevance, managerial communication of research, and especially evaluation of the artifact regarding its practical benefit.

Financial service providers currently promote the standardization of processes and IT applications to reduce their IT costs significantly. Here, the migration of monolithic legacy systems and the introduction of SoA become more and more important (see Baskerville et al. 2010 for further objectives of SoA). Against this backdrop, a SoA was introduced by the considered bank to restructure the IT applications of its division “securities”. One major aim of this 18 month running project was to define widely standardized services which can be used for the transaction of different types of securities over various distribution channels (branch, call center, internet, etc.). In one of the first steps, the bank analyzed its existing IT functions as well as its processes (cf. also Heinrich et al. 2009) to identify and decompose functions that may be realized as services. For that purpose, a function-oriented procedure was applied which results in a FSG (see Figure 2) like defined in section 3.

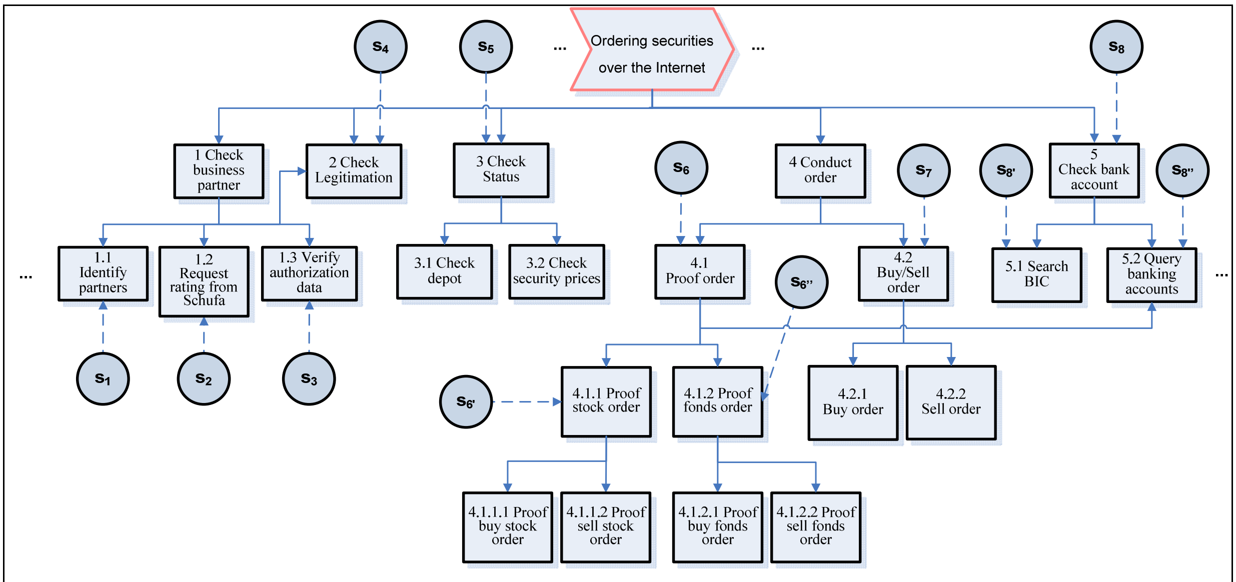

Figure 2. Extract of the FSG in the real world case of the financial service provider

By this procedure, we primarily analyzed which (decomposed) functions can be used for different types of securities and distribution channels and shall therefore be realized as a service (as function-oriented procedures are not in the focus of this paper, we do not go into more detail at this point). Two alternative, potential service realizations resulted from this analysis. In order to compare and discuss both service realizations, their particular granularities were measured amongst other things. Exemplarily, we focus on an extract of the security ordering process (“Ordering securities over the Internet”) that is illustrated in Figure 2. This process is decomposed in five functions: “Check business partner” comprises the functions “Identify partners”, “Request rating from Schufa” and “Verify authorization data”; “Check legitimation” is not further decomposed; “Check status” includes the functions “Check depot” and “Check security prices”;

"Conduct order" comprises the functions "Proof order" and "Buy/Sell order" which are both further decomposed; "Check bank account" includes the functions "Search BIC" and "Query banking accounts". As shown in Figure 2, functions like "Query banking accounts" and "Check legitimation" are used twice, which means each of them has two preceding functions (or a function and a process).To determine the size of already realized functions we could fall back on estimations of the IT department which are based on a software size measurement framework like the one proposed by Park (1992). The resulting LOC for each function are summarized in Table 4 where the values of the preceding functions comprise only the composition specification (the functions are referenced by the introduced number in Figure 2).

Table 4. Sizes of the functions in the real world case

function	$size_{m_f}$	function	$size_{m_f}$	function	$size_{m_f}$	function	$size_{m_f}$
1	200	*3.1*	1,600	*4.1.1.2*	600	*4.2.2*	800
1.1	1,500	*3.2*	200	*4.1.2*	200	*5*	400
1.2	600	*4*	300	*4.1.2.1*	600	*5.1*	1,000
1.3	1,000	*4.1*	400	*4.1.2.2*	600	5.2	1,400
2	1,400	*4.1.1*	200	*4.2*	450		
3	400	*4.1.1.1*	600	*4.2.1*	1,000		

Figure 2 also illustrates both alternative service realizations SR1 (including the services s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) and SR2 (including the services s_1, s_2, s_3, s_4, s_5, $s_{6'}$, $s_{6''}$, s_7, $s_{8'}$, $s_{8''}$). Using the presented metrics, we are now able to calculate the granularities for the single services and for the service realizations SR1 and SR2 (see Table 5).

Table 5. Calculated granularity values of the real world case

	Width metric	Depth metric	Combined width and depth metric	Size metric
Service s_1	1.00	1.00	1.00	0.12
Service s_2	1.00	1.00	1.00	0.25
Service s_3	1.00	1.00	1.00	0.16
Service s_4	1.00	1.00	1.00	0.13
Service s_5	0.50	0.00	0.00	0.00
Service s_6	0.20	0.33	0.13	0.06
Service $s_{6'}$	0.50	0.66	0.50	0.33
Service $s_{6''}$	0.50	0.66	0.50	0.33
Service s_7	0.50	0.50	0.33	0.12
Service s_8	0.50	0.00	0.00	0.00
Service $s_{8'}$	1.00	1.00	1.00	0.29
Service $s_{8''}$	1.00	1.00	1.00	0.33
Service Realization SR1	0.57	0.60	0.56	0.11
Service Realization SR2	0.71	0.78	0.73	0.20

In a first step, the services s_6 and s_8 in SR1 are of interest, since they are not part of SR2. Considering service s_6 and its granularity value for each metric, it becomes obvious that this service realizes neither a basic function nor a function that follows directly after a process. From the value of the depth metric results that the function realized by s_6 is located after a third of the complete path. The width metric illustrates that s_6 realizes five basic functions which are much more than other services. If we compare service s_6 with the alternative realization in SR2, i.e. the services $s_{6'}$, $s_{6''}$, and $s_{8''}$, it is immediately obvious that $s_{8''}$ is a basic function and that the services $s_{6'}$ and $s_{6''}$ realize two basic functions each which can be concluded from the granularity values of 0.5 of the width metric. In addition, the granularity value of 0.66 of the depth metric shows that - according to the complete path - the functions realized by the services $s_{6'}$ and $s_{6''}$ are located after the function realized by service s_6.

Moreover, an analysis of the granularity value of service s_8 - according to the depth metric as well as the combined width and depth metric - shows that this service realizes a function following directly after the process. The width metric specifies via a granularity value of 0.5 that the service s_8 realizes indirectly only two basic functions. A combined interpretation of both granularity values allows us to understand the structure of the sub-graph, where service s_8 is located. In this respect, the alternative services $s_{8'}$ and $s_{8''}$ and their granularity values are interpretable as well. Here, the first three metrics in Table 5 calculate a value of one, which means, services $s_{8'}$ and $s_{8''}$ realize basic functions. Overall, this analysis shows that the considered sub-graph consists of a function, which follows directly after the process and which is decomposed into two basic functions.

Considering in a second step the overall granularity values of SR1 and SR2 (cf. the last two columns of Table 5), it becomes obvious that (according to all metrics) SR2 is more fine-grained than SR1, which was finally chosen by the bank. This illustrates that several alternative service realizations can be easily compared using the metric values of the overall granularity, especially in the case of a large and complex FSG.

Based on the two alternative service realizations, we may further illustrate the economic effects of the granularity choice. As already discussed above, the choice of service granularity may influence three cost components of a service realization: 1) implementation costs, 2) maintenance costs, and 3) composition costs (according to Krammer 2011).

Ad 1) Coarse-grained service realizations usually lead to higher implementation costs compared to fine-grained service realizations in case both realizations implemented the same (sizes of the) functions. This is due to the fact that the increasing size of a service and the corresponding higher complexity leads to a disproportional increase of the implementation and testing effort. For instance, the implementation costs for service s_8 within SR1 are higher than the sum of the implementation costs for the services $s_{8'}$ and $s_{8''}$ within SR2. Moreover, the covered functions by a coarse-grained service may have to be realized several times. This can be illustrated by our real world case. Choosing service realization SR1, service $s_{8''}$ cannot be reused and thus function “5.2 Query banking accounts” has to be realized twice (once by service s_6 and once by service s_8). Thus the implementation costs of the more coarse-grained service realization SR1 (regarding all granularity metrics) are higher compared to service realization SR2.

Ad 2) Coarse-grained service realizations usually lead to higher maintenance costs compared to fine-grained service realizations, too. Maintenance costs depend also on the size of a service and the corresponding higher complexity which has to be handled when maintaining a service. Thus, not only the implementation costs for service s_8 within SR1 are higher than the sum of the implementation costs for the services $s_{8'}$ and $s_{8''}$ within SR2 (cf. example in Ad 1)) but also the maintenance costs. Moreover, the maintenance costs depend on the number of redundantly realized functions because adaptations of functions have to be made several times. When choosing service realization SR1 in our real world case, potentially required adaptations of the function “5.2 Query banking accounts” would have to be made not only for one service but for the two services s_6 and s_8. Thus, the maintenance costs of service realization SR1 are higher compared to service realization SR2.

Ad 3) Composition costs of a service realization result from the effort to realize the composition specification (e.g. by creating a WS-BPEL file) for searching and integrating single services. This effort mainly depends on the number of realized functions which have to be composed to execute a process. Regarding our real world case, the composition costs for service realization SR1 are lower than for service realization SR2 as for SR1 eight services (s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8) and for SR2 ten services (s_1, s_2, s_3, s_4, s_5, $s_{6'}$, $s_{6''}$, s_7, $s_{8'}$, $s_{8''}$) have to be composed to execute the loan process.

To sum up, we can state an economic trade-off resulting from the chosen service granularity between implementation and maintenance costs on the one hand and composition costs on the other hand.

This brief discussion demonstrates not only the economic trade-off but also motivates the importance of measuring service granularity in practice.

Conclusion

In this paper, we presented different metrics which support a comprehensible measurement of service granularity related to the seven guidelines for conducting Design Science Research by Hevner et al. (2004) we can conclude as follows:

Our developed *artifacts* (cf. Guideline 1) are four metrics to measure the granularity for IT services. Statements in literature support that choosing the "right" granularity is a highly *relevant* problem when designing a SoA (cf. Guideline 2). This is due to the fact that service granularity influences important aspects like the reuse of services or the performance of the execution of a service composition and thus the implementation, maintenance, and composition costs. Moreover in research and practice, there is a lack of granularity metrics for IT services which consider criteria emphasized in literature (*number* and *size* of the realized functions, *decomposition layer* where a realized function is located according to a formal function tree, *normalization* of the values of the metrics). By means of these criteria, we presented four different metrics. Furthermore, we analyzed their particular advantages and disadvantages, discussed their interpretability and perspectives on service granularity and gave some practical hints to the use of them. Furthermore, we *evaluated* (cf. Guideline 3) our metrics regarding their applicability and their practical utility provided in a real world case of a major German bank. Thus, the metrics were analyzed "in depth in business" (Hevner et al. 2004). We regard the developed metrics as an important step to solve the key problem of measuring service granularity. More precisely, our metrics *contribute to the research* (cf. Guideline 4) by making an evaluation and comparison of the granularity of various service realizations possible. Furthermore, they may constitute a first step towards the evaluation of the implementation, maintenance and composition costs of service realizations. To assure a *rigorous* (cf. Guideline 5) definition and presentation, we denoted the metrics formally using mathematical expressions.

Regarding the *search process* (cf. Guideline 6), present and future steps can be distinguished. In this paper, the design process was guided by the mentioned criteria in the existing literature. To illustrate our design steps we described alternative metrics (cf. footnotes 4-7) and discussed their disadvantages to the proposed metrics. To develop these granularity metrics, we used a graph definition - which is largely inspired by the well-known approach of Lee et al. (2001) - in order to complement existing function-oriented approaches. In this way, we minimized not only the subjective modelers' preferences regarding the creation of a particular graph but also ensured a high significance of the service granularities measured by our metrics. Since other authors of further function-oriented procedures specify similar graph definitions (e.g. Wang et al. 2006; Winkler 2007), our granularity metrics are also applicable based on those approaches (i.e. the metrics are not limited to one particular graph definition).

In future steps we want to provide - as mentioned above - a well-founded evaluation of different existing procedures for service identification comparing the granularity of the services identified by each particular procedure. This would help to find deeper insights from a Design Science perspective whether one of these procedures leads in general to more fine- or coarse-grained services than another procedure. This seems not only interesting from a scientific point of view but also for companies that adapt and use such procedures in practice.

In addition, the presented metrics have to be extensively evaluated based on mathematical properties (cf. e.g. Briand et al. 1996; Habra et al. 2008; Poels and Dedene 2000; Weyuker 1988). For instance, Briand et al. (1996) developed sets of properties for different types of software metrics and evaluated several existing metrics against these properties. They defined different sets of properties for the measurement concepts *size, length, complexity, cohesion,* and *coupling*. For our metrics, the measurement concepts *size, length,* and *complexity* are relevant (we delimited our research from works considering cohesion and coupling metrics already in the introduction).

For instance, for the concept *size*, Briand et al. (1996, p. 71-72) propose the properties "Nonnegativity", "Null value", and "Module Additivity" to evaluate whether or not a given metric characterize the size of a system: Regarding our size metric denoted by Term (4), the property "Nonnegativity" holds for $size_{m_a}^{agg} \geq size_{m_j}^{agg} > 0$. This is true as basic and preceding functions cannot have a negative size $size_{m_j}^{agg}$. The property "Null Value" expresses that the size of a system is zero if $E=\varnothing$ holds (i.e. $\nexists\, (s_i, m_j) \in E$). In this case, the value z_s of our size metric is defined to be zero (see above). Therefore, our size metric meets the property "Null Value" as well. Last, the property "Module Additivity" means that "the size of a system is equal to the sum of the sizes of two of its modules such that any element of the system is an element of exactly one of its two modules" (Briand et al. 1996, p. 71). We can analyze this property by a simple case. A FSG which meets the condition that any element (e.g. a function measured by LOC) of the FSG is an element of exactly one of its modules, is represented by a process p that is followed directly by two (or more) realized basic functions m_j.[7] In this case, each function m_j equals m_a and consequently both values of the size metric z_s equals zero (cf. Term (4)). Here, the overall granularity of a system (i.e. a FSG consisting of two or more maximal coarse-grained services) is equal to the sum of the granularity of two (or more) of its modules (i.e. services). Thus, the property "Module Additivity" holds as well. In addition to this argumentative evaluation of the size metric, we need to do a mathematical proof of the three properties in a next step. Furthermore, we need to mathematically evaluate the defined sets of properties (cf. Briand et al. 1996) for the other presented metrics in further research (due to length restrictions these extensive mathematical proofs cannot be included in this paper). Based on the results of these evaluations, we may need to adapt the metrics and have to critically analyze their feasibility.

Moreover, we are currently working on quantitative models based on our granularity metrics to determine optimal service realizations of different FSGs which minimize the sum of implementation, maintenance and composition costs (for a first step towards such a quantitative model, cf. Krammer 2011). To evaluate the models we aim to use empirical data of the major German bank of our real world case. Thus, according to the Goal-Question-Metric paradigm (Basili et al. 1994), our *metrics* represent an important first step to answer the *question* of how to adequately choose the service granularity and to reach the *goal* of an efficient support of business processes by IT services.

Finally, for the *communication* (cf. Guideline 7) of our research, in most parts of the paper we chose a formal, mathematical presentation to be able to demonstrate our artifacts in a rigorous way. However, we also tried to attract a managerial audience by means of the extensive illustration and discussions of the metrics as well as the detailed demonstration of their application and their practical utility.

Acknowledgements

The research was funded by the Austrian Science Fund (FWF): P 23546-G11 and P23567-G11.

[7] In case of more than one decomposition layers, the size of the basic functions m_j is represented by more than one element (i.e. function) of the FSG.

References

Aier, S. 2006. "How Clustering Enterprise Architectures helps to Design Service Oriented Architectures," in *Proceedings of the IEEE International Conference on Services Computing*, Chicago, IL, pp. 269-272.

Alahmari, S., De Roure, D., and Zaluska, E. 2010. "A Model-Driven Architecture Approach to the Efficient Identification of Services on Service-oriented Enterprise Architecture," in *Proceedings of the Workshop on Service oriented Enterprise Architecture for Enterprise Engineering*, Vitória, Brazil, (available online at http://eprints.ecs.soton.ac.uk/21359/1/A_Model-Driven_Architecture_Approach_to_the_Efficient_Identification_of_Services_on_Service-oriented_Enterprise_Architecture.pdf).

Albani, A., Overhage, S., and Birkmeier, D. 2008. "Towards a Systematic Method for Identifying Business Components," in *Proceedings of the Symposium on Component-Based Software Engineering*, Leipzig, Germany, pp. 262-277.

Basili, V. R., Caldiera, G., and Rombach, H. D. 1994. "Goal Question Metric Paradigm," *Encyclopedia of Software Engineering*, Hoboken, NJ: John Wiley & Sons, pp. 528–532.

Baskerville, R. L., Cavallari, M., Hjort-Madsen, K., Pries-Heje, J., Sorrentino, M., and Virili, F. 2010. "The strategic value of SOA: a comparative case study in the banking sector," *International Journal of Information Technology and Management* (9:1), pp. 30-53.

Boehm, B. W., Abts, C., Windsor Brown, A., Chulani, S., Clark, B. K., Horowitz, E., Madachy, R., Reifer, D., and Steece, B. 2000. *Software Cost Estimation with COCOMO II*, Upper Saddle River, NJ: Prentice Hall.

Briand L., Morasca S. and Basili V. 1996. "Property-Based Software Engineering Measurement," *IEEE Transactions on Software Engineering* (22:6), pp. 68-86.

Buhl, H. U., Heinrich, B., Henneberger, M., Alexander Krammer, A. 2008. "Service Science," *Wirtschaftsinformatik/Business & Information Systems Engineering* (50:1), pp. 60-65 (in German).

Chidamber, S. R., and Kemerer, C. F. 1994. "A Metrics Suite for Object Oriented Design," *IEEE Transactions on Software Engineering* (20:6), pp. 476-493.

Counsell, S., Swift, S., and Crampton, J. 2006. "The interpretation and utility of three cohesion metrics for object-oriented design," *ACM Transactions on Software Engineering and Methodology* (15:2), pp. 123-149.

Dolado, J.J. 2000. "A validation of the component-based method for software size estimation," *IEEE Transactions on Software Engineering* (26:10), pp. 1006-1021.

Erl, T. 2007. *SOA: Principles of Service Design*, Upper Saddle River, NJ: Prentice Hall.

Erl, T. 2004. *Service-Oriented Architecture - A Field Guide to Integrating XML and Web Services*, Upper Saddle River, NJ: Prentice Hall.

Feuerlicht, G. 2011. "Simple Metric for Assessing Quality of Service Design," *Service-Oriented Computing, Lecture Notes in Computer Science* (6568), pp. 133-143.

Fiege, R., and Stelzer, D. 2007. "Axiomatic Design to Model Service Oriented Architectures," in *Proceedings of the 8th International Conference on Business Informatics*, Karlsruhe, Germany, pp. 909-926 (in German).

Galster, M., and Bucherer, E. 2008. "A Business-Goal-Service-Capability Graph for the Alignment of Requirements and Services," in *Proceedings of the IEEE Congress on Services*, Honolulu, Hawaii, pp. 399-406.

Habra, N., Abran, A., Lopez, M., and Sellami, A. 2008. "A framework for the design and verification of software measurement methods," *The Journal of Systems and Software* (81:5), pp. 633-648.

Haesen, R., Snoeck, M. Lemahieu, W., and Poelmans, S. 2008. "On the Definition of Service Granularity and its Architectural Impact, *Lecture Notes in Computer Science* (5074), pp. 375-389.

Heinrich, B., Henneberger, M., Leist, S. Zellner, G. 2009. "The Process Map as an Instrument to Standardize Processes: Design and Application at a Financial Service Provider," *Information Systems and e-Business Management* (7:1), pp. 81-102.

Her, J. S., La, H. J. and, Kim, S. D. 2008. "A Formal Approach to Devising a Practical Method for Modeling Reusable Services," in *Proceedings of the International Conference on e-Business Engineering*, Xi'an, China, pp. 221-228.

Hevner, A. R., March, S. T., Park, J., and Ram S. 2004. "Design Science in Information Systems Research," *MIS Quarterly* (28:1), pp. 75-105.

Khoshkbarforoushha, A., Tabein, R., Jamshidi, P., and Shams, F. 2010. "Towards A Metrics Suite for Measuring Composite Service Granularity Level Appropriateness," in *Services '10 Proceedings of the 6th World Congress on Services*, Washington, DC: IEEE Computer Society, pp. 245-252.

Krafzig, D., Banke, K., and Slama, D. 2005. *Enterprise SOA – Service-Oriented Architecture Best Practices*, Upper Saddle River, NJ: Prentice Hall.

Krammer, A., Heinrich, B., Henneberger, M. and Lautenbacher, F. 2011. „Granularity of services - an economic analysis," *Business & Information Systems Engineering* (3:6), pp. 345-358.

Lee, J. K., Jung, S. J., Kim, S. D., Jang, W. H., Ham, D. H. 2001. „Component Identification Method with Coupling and Cohesion," *Proceedings of the Asia-Pacific Software Engineering Conference*, Macau, China, pp. 79-87.

Ma, Q., Zhou, N., Zhu, Y., and Wang, H. 2009. „Evaluating Service Identification with Design Metrics on Business Process Decomposition," in *Proceedings of the IEEE International Conference on Services Computing*, Bangalore, India, pp. 160-167.

Marks, E. A., and Bell, M. 2006. *Service-Oriented Architecture: A Planning and Implementation Guide for Business and Technology*. Hoboken, NJ: Wiley.

Papazoglou, M. P. 2003. "Service-Oriented Computing: Concepts, Characteristics and Directions," in *Proceedings of the International Conference on Web Information Systems Engineering*, Rome, Italy, pp. 3-12.

Papazoglou, M. P., and van den Heuvel, W. J. 2006. "Service-Oriented Design and Development Methodology," *International Journal of Web Engineering and Technology* (2:4), pp. 412-442.

Papazoglou, M. P., Traverso, P., Dustdar, S., Leymann, F. and Krämer, B. J. 2008. "Service Oriented Computing: A Research Roadmap," *International Journal of Cooperative Information Systems* (17:2), pp. 223-255.

Park, R. E. 1992. "Software Size Measurement: A Framework for Counting Source Statements," *Technical Report*, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, (available online at http://www.sei.cmu.edu/reports/92tr020.pdf).

Poels G. and Dedene G. 2000. "Distance-based Software Measurement: Necessary and Sufficient Properties for Software Measures," *Information and Software Technology* (42:1), pp. 35-46.

Perepletchikov, M., and Ryan, C. 2011. "A Controlled Experiment for Evaluating the Impact of Coupling on the Maintainability of Service-Oriented Software," *IEEE Transactions on Software Engineering* (37:4), pp. 449-465.

Perepletchikov, M., Ryan, C., and Frampton, K. 2007a. "Cohesion Metrics for Predicting Maintainability of Service-Oriented Software," in *Proceedings of the 7th International Conference on Quality Software, IEEE Computer Society*, pp. 328-335.

Perepletchikov, M., Ryan, C., Frampton, C. K., and Tari, Z. 2007b. "Coupling Metrics for Predicting Maintainability in Service-Oriented Designs," in *Proceedings of the Australian Software Engineering Conference, IEEE Computer Society*, pp. 329-340.

Perepletchikov, M., Ryan, C., and Tari. Z. 2010. "The Impact of Service Cohesion on the Analyzability of Service-Oriented Software," *IEEE Transactions on Services Computing* (3:2), pp. 89–103.

Schelp, J., and Winter, R. 2008. "Application Design and Enterprise Service Design – Similarities and Differences," *Wirtschaftsinformatik/Business & Information Systems Engineering* (50:1), pp. 6-15 (in German).

Sims, O. 2005. "Developing the architectural framework for SOA - part 2 - service granularity and dependency management," *CBDI Journal* (6).

Wang, Z., Xu, X., and Zhan, D. 2005. "A Survey of Business Component Identification Methods and Related Techniques," *International Journal of Information Technology* (2:4), pp. 229-238.

Wang, Z., Zhan, D. and Xu, X. 2006. "STCIM: a dynamic granularity oriented and stability based component identification method," *ACM SIGSOFT Software Engineering Notes* (31:3), pp. 1-14.

Webster, J., and Watson, R. T. 2002. "Analyzing the Past to Prepare for the Future: Writing a Literature Review," *MIS Quarterly* (26:2), pp. xiii-xxiii.

Weyuker, E.J. 1988. "Evaluating software complexity measures," *IEEE Transactions on Software Engineering* (14:9), pp. 1,357-1,365.

Winkler, V. 2007. "Identification and Design of Services – Proceeding and Exemplary Use in the Financial Services Sector," *Wirtschaftsinformatik/Business & Information Systems Engineering* (49:4), pp. 257-266 (in German).

Winter, R. 2003. "An Architecture Model for Supporting Application Integration Decisions," in: *Proceedings of the 11th European Conference on Information Systems*, Naples, Italy (available online at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.99.1625&rep=rep1&type=pdf)

Zhang, Q., and Li, X. 2009. "Complexity Metrics for Service-Oriented Systems," in *Proceedings of the International Symposium on Knowledge Acquisition and Modeling*, Huazhong, China, pp. 375-378.